CSE210

Gavin Hart

Goals:

* Explain the meaning of Abstraction.
* Highlight a benefit of Abstraction.
* Provide an application of Abstraction.
* Use a code example of Abstraction from the program you wrote.
* Thoroughly explain these concepts (this likely cannot be done in less than 100 words)

Abstraction means to take a complex idea or concept and simplify it into something more manageable by only looking at the most essential parts. It's like looking at the big picture without getting bogged down in the small details. Abstraction is a fundamental concept in many areas of human activity, including art, science, mathematics, and computer programming.

The benefits of abstraction are numerous Like I said above it can be used:

1. Simplification: Abstraction helps to simplify complex ideas or systems, making them more manageable and easier to understand.

2. Efficiency: Abstraction can lead to more efficient use of resources by focusing only on the essential aspects of a system or problem.

3. Flexibility: Abstraction allows for greater flexibility in problem-solving by enabling us to think about problems in different ways and from different perspectives.

4. Reusability: Abstraction can make it easier to reuse solutions to similar problems, saving time and effort in the long run.

Overall, abstraction is a powerful tool that can help us to solve things. One application of abstraction is in computer programming, where it allows developers to create complex software systems by breaking them down into smaller, more manageable parts. By abstracting away the details of how each part works, developers can create a more modular and reusable codebase that is easier to maintain and update over time.

class Animal:

def \_\_init\_\_(self, name):

self.name = name

def speak(self):

pass

class Dog(Animal):

def speak(self):

return "Woof!"

class Cat(Animal):

def speak(self):

return "Meow!"

def animal\_speak(animal):

print(animal.speak())

dog = Dog("Fido")

cat = Cat("Whiskers")

animal\_speak(dog) # Output: "Woof!"

animal\_speak(cat) # Output: "Meow!"

In this example, we have an Animal class that serves as an abstraction for different types of animals. The Dog and Cat classes are both derived from the Animal class, but they implement their own version of the speak () method, which returns the sound that each animal makes.

The animal speak () function takes an instance of the Animal class as an argument and calls its speak() method, which is implemented differently for each subclass. By abstracting away the details of how each animal speaks, we can write a more general function that works for any type of animal without having to know the specifics of each one.

This is something I have used in interviews for jobs and past careers I have had before changing my career choice. In all abstraction is a way of communication and is used for a lot of avenues.